Agenda Con Fragments

MainActivity:

En la actividad principal tengo los datos que paso a mi fragmento

```
fragmentLista = new MiFragmento(datos, imagenPorDefecto);
FragmentManager fragmentManager = getSupportFragmentManager();
FragmentTransaction fragmentTransaction = fragmentManager.beginTransact
fragmentTransaction.add(R.id.fragment_container, fragmentLista);
fragmentTransaction.commit();
```

Y tengo dos métodos estáticos que convierte una imagen bitmap a String, y otro método que convierte String a bitmap para pasar a otra activity desde la primera activity.

```
public static String BitmapAString(Bitmap imagen) {
    ByteArrayOutputStream stream = new ByteArrayOutputStream();
    imagen.compress(Bitmap.CompressFormat.PNG,  quality: 90, stream);
    byte[] byte_arr = stream.toByteArray();
    String image_str = Base64.encodeToString(byte_arr, Base64.DEFAULT);
    return image_str;
}
public static Bitmap StringABitmap(String imagen) {
    byte[] decodedString = Base64.decode(imagen, Base64.DEFAULT);
    return BitmapFactory.decodeByteArray(decodedString,  offset: 0, decodedSt
}
```

MiFragmento:

Tengo un Constructor que recibe datos y asigna los datos.

```
public MiFragmento(ArrayList<Persona> lista, String imagenPorDefecto){
    datos = lista;
    this.imagenPorDefecto = imagenPorDefecto;
}
```

Y en onCreateView asigno el rootView a la actividad de recycler view.

Cojo las imagenes y todos los botones que tenga el recyclerView. Por ejemplo el fab:

```
imagen = rootView.findViewById(R.id.imageView);

FloatingActionButton fab = rootView.findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        AñadirDatos();
    }
});
```

Busco y coloco el recyclerView en la clase, y creo su adaptador y el holder para recyclerView.

En el **Adaptador**:

implementamos Listeners → click, onLongClick y El swipe que será de touch. Y creamos el **holder**: la vista que queremos que se aparezca de cada item.

```java
public class Holder extends RecyclerView.ViewHolder {

    Context context;
    View itemView;

    TextView txtNombre, txtApellido, txtTelefono, txtCorreo;
    ImageView imagen;
    View.OnClickListener listener;

    public Holder(View itemView, Context context) {
        super(itemView);
        this.context = context;
        this.itemView = itemView;
        txtNombre =  itemView.findViewById(R.id.Nombre);
        txtApellido= itemView.findViewById(R.id.Apellidos);
        txtCorreo = itemView.findViewById(R.id.Correo);
        txtTelefono = itemView.findViewById(R.id.Telefono);
        imagen = itemView.findViewById(R.id.imageView);

    }
```

Buscamos y asignamos los valores.

```java
public  void bind(Persona persona, int pos){

    txtNombre.setText(persona.getNombre());
    txtApellido.setText(persona.getApellidos());
    txtCorreo.setText(persona.getCorreo());
    txtTelefono.setText(persona.getTelefono());
    imagen.setImageBitmap(MainActivity.StringABitmap(persona.getImagen()));
}
```

Ahora el objeto que le pasamos, asignamos valores.

```java
public Adaptador(Context c) { this.c = c; }

@Override
public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int vie
    View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.c
    view.setOnTouchListener(this);
    view.setOnClickListener(this);
    view.setOnLongClickListener(this);
    h = new Holder(view,c);

    return h;
}
```

En el adaptador inflatamos la vista y asignamos los Listeners.

```java
@Override
public void onBindViewHolder(RecyclerView.ViewHolder holder, int position)
    ((Holder)holder).bind(((MainActivity)c).datos.get(position), position)
}
```

Saca un datos de la posicion.

```java
@Override
public void onClick(View view) { if(listener!=null) listener.onClick(view


public void SetOnTouchListener(View.OnTouchListener touchListener) {
    if(touchListener!=null) onTouchListener = touchListener;
}
@Override
public boolean onTouch(View view, MotionEvent motionEvent) {
    if(onTouchListener != null) {
        onTouchListener.onTouch(view, motionEvent);
    }
    return false;
}

public void  SetOnLongClick(View.OnLongClickListener longClickListener) {
    if(longClickListener != null) {
        this.longClickListener = longClickListener;
    }
```

Implementamos los listeners..

```java
    recyclerView = rootView.findViewById(R.id.recycler);
    recyclerView.setHasFixedSize(true);
    adaptador = new Adaptador(getContext());
    recyclerView.setAdapter(adaptador);
    recyclerView.setLayoutManager(new LinearLayoutManager(getActivity(),
    adaptador.SetOnLongClick( longClickListener: this);
    swipeDetector = new SwipeDetector();
    adaptador.setClickOnView(this);
    adaptador.SetOnTouchListener(swipeDetector);

    return rootView;
}
```

Al final en mi fragmento, Asigno cada valor al RecyclerView, y le asigno el adaptador. Y al final retorno a la vista que esta obligado en su interfaz.

En mi fragmento creo onlongClick:Para que si le hago onLongClick me abre un dialogo, àra eliminar o dar se a cancelar.

```java
@Override
public boolean onLongClick(View v) {
    pos = recyclerView.getChildAdapterPosition(v);
    Persona d = datos.get(pos);
    AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
    builder.setMessage("¿Seguro que quieres eliminar a " + d.getNombre()
    builder.setPositiveButton( text: "ELIMINAR", (dialog, which) → {
            datos.remove(pos);
            recyclerView.setAdapter(adaptador);
    });
    builder.setNegativeButton( text: "CANCELAR", (dialog, which) → {
            dialog.cancel();
    });
    builder.create().show();
    return true;
}
```

En onClick asigno primero la posicion de la lista, al que he hecho el click.

```
@Override
public void onClick(View v) {
    pos = recyclerView.getChildAdapterPosition(v);
    if (swipeDetector.swipeDetected()) {
        AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
        switch (swipeDetector.getAction()) {
            case LR:
                builder.setMessage("¿Estas seguro que quieres llamar a " + datos.get(pos).getNombre() + "?");
                builder.setPositiveButton( text: "LLAMAR", (dialog, which) → {
                        Intent intent = new Intent(Intent.ACTION_DIAL);
                        intent.setData(Uri.parse("telefono:" + datos.get(pos).getTelefono()));
                        if (intent.resolveActivity(getContext().getPackageManager()) != null) {
                            startActivity(intent);
                        }
                });
                builder.setNegativeButton( text: "Cancelar", (dialog, which) → {
                        dialog.cancel();
                });
                builder.create().show();
                break;
            case RL:
                builder = new AlertDialog.Builder(getContext());
                builder.setMessage("¿Estas seguro que quieres enviar mensaje a " + datos.get(pos).getNombre() + "?");
                builder.setPositiveButton( text: "enviar", (dialog, which) → {
                        Intent intent = new Intent(Intent.ACTION_SENDTO);
                        intent.setData(Uri.fromParts( scheme: "mailto", datos.get(pos).getCorreo(), fragment: null));
                        Intent chooser = Intent.createChooser(intent, title: "Enviar mensaje...");
                        startActivity(chooser);
                });
                builder.setNegativeButton( text: "Cancelar", (dialog, which) → {
                        dialog.cancel();
                });
                builder.create().show();
```

Y si es un swipe Izquierda a derecha. Se le abre un diálogo para llamar al teléfono. Y si damos a aceptar que nos lleva a las llamadas para llamar con su número.
Y si es un swipe de derecha a Izquierda, abre un diálogo para enviar un correo, y si le damos aceptar no lleva a gmail para enviar el correo.
Y si no se detecta ningún swipe se abre una actividad de Editar datos:

```
        }
    } else {
        EditarDatos();
    }
```

```
}
private void EditarDatos() {
    Intent intent = new Intent(getContext(), EditarContacto.class);
    Persona p = datos.get(pos);
    intent.putExtra( name: "DatoPersona", p);
    startActivityForResult(intent,CODIGO_EDITAR);
}
private void AñadirDatos() {
    Intent intent = new Intent(getContext(), EditarContacto.class);
    Persona p = null;
    intent.putExtra( name: "DatoPersona", p);
    startActivityForResult(intent,CODIGO_AÑADIR);
}
```

Como habiamos visto en el onClick del botón “fab” llamabamos al método AñadirDatos(),
Pués era simplemente crear un intent y poner un put extra de un objeto parcelable y iniciar la actividad con el codigo de añadir.
Y en editar contacto hacemos lo mismo pero pasando un objeto de la posición seleccionada.

Y Para editar o añadir el contacto tengo una activity en el que se edita y una clase que se llama a esa activity para obtener los datos de la activity.

```
public class EditarContacto extends AppCompatActivity implements View.OnClickLi

    private static final int CODIGO_GALERIA = 1;

    private Persona p = null;
    private ImageView imagen;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_edit_contact);

        final EditText editTextNombre = findViewById(R.id.editText);
        final EditText editTextApellidos = findViewById(R.id.editText2);
        final EditText editTextTelefono = findViewById(R.id.editText3);
        final EditText editTextCorreo = findViewById(R.id.editText4);
        imagen = findViewById(R.id.imagen);
        imagen.setOnClickListener(this);
        FloatingActionButton fab = findViewById(R.id.floatingActionButton);

        Intent intentR = getIntent();
        p = intentR.getParcelableExtra( name: "DatoPersona");
```

Si el intento es nulo creamos un nuevo objeto y le pasamos y si no, pasamos los datos desde la clase parcelable.

```
Intent intentR = getIntent();
p = intentR.getParcelableExtra( name: "DatoPersona");

if(p!= null){

    editTextNombre.setText(p.getNombre());

    editTextApellidos.setText(p.getApellidos());

    editTextTelefono.setText(p.getTelefono());

    editTextCorreo.setText(p.getCorreo());

    String ArrayImagen = p.getImagen();
    imagen.setImageBitmap(MainActivity.StringABitmap(ArrayImagen));

}
else {
    p = new Persona();
}
```

Tiene un floating action button para guardar los datos y enviar los.
Asignamos los datos a la clase parcelable, lo que cambiamos en el textView de la actividad.

```
etOnClickListener((v) → {

    Intent intent = new Intent(getApplicationContext(), MainActivity.class);
    p.setNombre(editTextNombre.getText().toString());
    p.setApellidos(editTextApellidos.getText().toString());
    p.setCorreo(editTextCorreo.getText().toString());
    p.setTelefono(editTextTelefono.getText().toString());

    BitmapDrawable drawable = (BitmapDrawable) imagen.getDrawable();
    if(drawable != null) {
        Bitmap bitmap = drawable.getBitmap();
        p.setImagen(MainActivity.BitmapAString(bitmap));
    }
    else
        p.setImagen(MainActivity.BitmapAString(BitmapFactory.decodeResource(Edi


    if(!p.getNombre().equals("")){
        intent.putExtra( name: "PersonaEditado", p);
        setResult(RESULT_OK, intent);
        finish();
    }
    else
        Toast.makeText(getApplicationContext(), text: "El nombre es obligatorio"
```

Al final le pasamos el objeto parcelable con una etiqueta, y terminamos la actividad.

En la clase EditarContacto implementamos onClick y abrimos la galeria para coger la foto y implementamos OnActivityResult para eso.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    Uri selectedImage;

    switch (requestCode) {
        case CODIGO_GALERIA:
            if (resultCode == RESULT_OK) {
                selectedImage = data.getData();
                String selectedPath = selectedImage.getPath();
                if (requestCode == 1) {
                    if (selectedPath != null) {
                        InputStream imageStream = null;
                        try {
                            imageStream = getContentResolver().openInputStream(
                                    selectedImage);
                        } catch (FileNotFoundException e) {
                            e.printStackTrace();
                        }
                        Bitmap bmp = BitmapFactory.decodeStream(imageStream);
                        imagen.setImageBitmap(Bitmap.createScaledBitmap(bmp,  dstWidth: 100,  dstHeight: 100,  filter: true));
                    }
                }
            }
            break;
    }
}
```

Cambio el tamaño de la imagen a 100x100. Para no tener problemas con la clase parcelable, porque no puedo enviar tamaño maximo pasado de los que esta fijado..
Para que se abriera esta actividad hay que poner en Androidmanifest.xml esto:

```
<activity
    android:name=".EditarContacto"
    android:label="Editar Contacto"
    android:theme="@style/AppTheme.NoActionBar">
</activity>
```

Y para recibir los datos en mi fragmento tengo que implementar OnActivityResult, Puedo elegir imagen desde la página principal.

```
switch (requestCode) {
    case CODIGO_GALERIA:
        if (resultCode == RESULT_OK) {
            selectedImage = data.getData();
            String selectedPath=selectedImage.getPath();
            if (requestCode == 1) {
                if (selectedPath != null) {
                    InputStream imageStream = null;
                    try {
                        imageStream = getContext().getContentResolver().openInputStream(
                                selectedImage);
                    } catch (FileNotFoundException e) {
                        e.printStackTrace();
                    }
                    Bitmap bmp = BitmapFactory.decodeStream(imageStream);
                    datos.get(pos).setImagen(BitmapAString(Bitmap.createScaledBitmap(bmp,  dstWidth: 100, dstHeight: 100,  filter: true)));
                }
            }
        }
        break;
```

Luego para Añadir y editar hay que rellenar los datos. Y muy importante:::: Cada vez que cambiamos algo en el recyclerView Hay que volver a setear el adaptador al recyclerView. Así se actualiza la lista.

```
            break;
        case CODIGO_EDITAR:
            if (resultCode == RESULT_OK) {
                p = data.getParcelableExtra( name: "PersonaEditado");
                if(p != null){
                    datos.get(pos).setTelefono(p.getTelefono());
                    datos.get(pos).setApellidos(p.getApellidos());
                    datos.get(pos).setCorreo(p.getCorreo());
                    datos.get(pos).setNombre(p.getNombre());
                    datos.get(pos).setImagen(p.getImagen());

                }
            }
            break;
        case CODIGO_AÑADIR:
            if (resultCode == RESULT_OK) {
                p = data.getParcelableExtra( name: "PersonaEditado");
                if(p != null){
                    p.setImagen(imagenPorDefecto);
                    datos.add(p);
                }
            }
            break;
    }
    recyclerView.setAdapter(adaptador);
}
```

En la actividad de recycler tengo solo, un floating action button, y Un recyclerView que se encarga de la lista de datos.

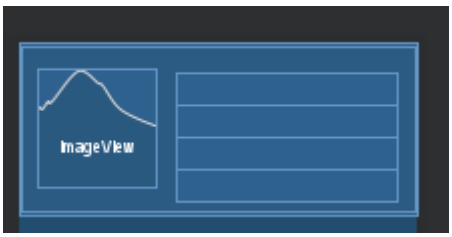
```
<androidx.coordinatorlayout.widget.CoordinatorLayout xmlns:android="http://schemas.android.com
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:layout_margin="16dp"
        android:backgroundTint="@color/colorPrimary"
        app:srcCompat="@android:drawable/ic_input_add" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/recycler"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginStart="8dp"
            android:layout_marginTop="8dp"
            android:layout_marginEnd="8dp"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
    </LinearLayout>
```

En actividad tengo un FrameLayout que es un contenedor de mi Fragmento.

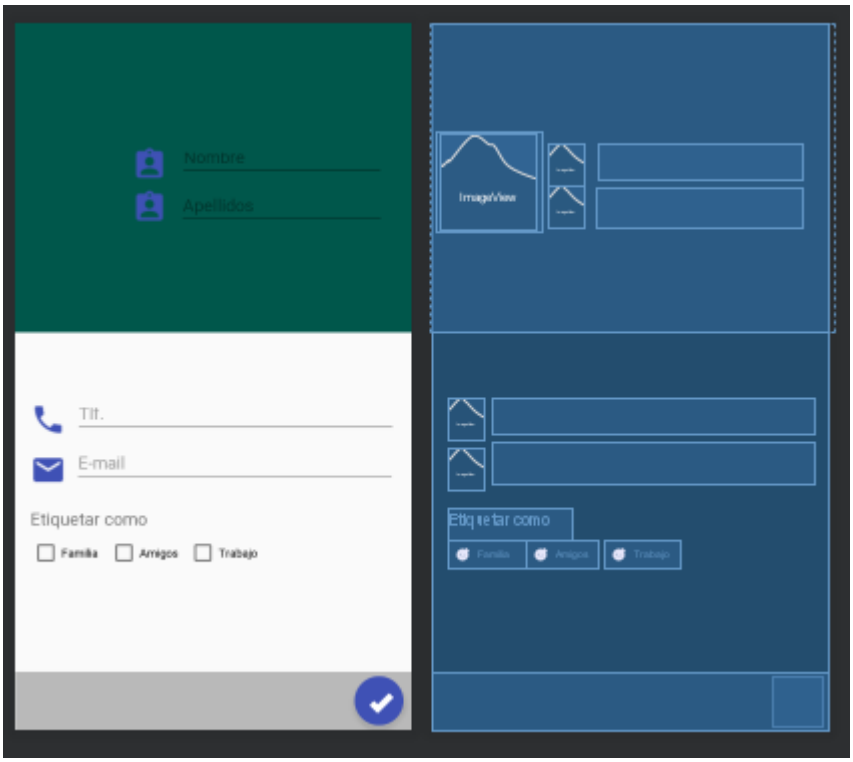
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:orientation="vertical">

    <FrameLayout
        android:id="@+id/fragment_container"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#3355CC00"
        />

</LinearLayout>
```

Tengo una actividad de contacto en el que edito todos los textView. Y la imagen. El especto de cada dato que aparecerá en la vista de android.

Es un cardView en el que tiene 4 textView y una imagen.



Y al final para editar contacto tengo la actividad: con el que cambio el objeto.



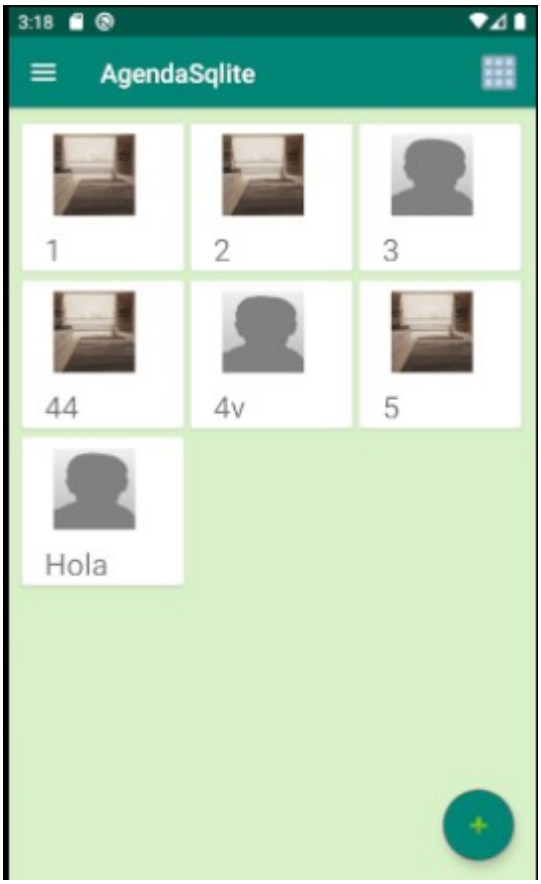
Así sería el aspecto…

19/01/2020:
Recordamos que hicimos la agenda con fragments. Ahora vamos a agregar los menus y le conectamos con la base de datos sqlite:

Vale empezamos tenemos que cambiar un poco de código en los apartados que voy a explicar ahora.

Voy a empezar con los menus:
después nuestra agenda tendrá un aspecto como este:

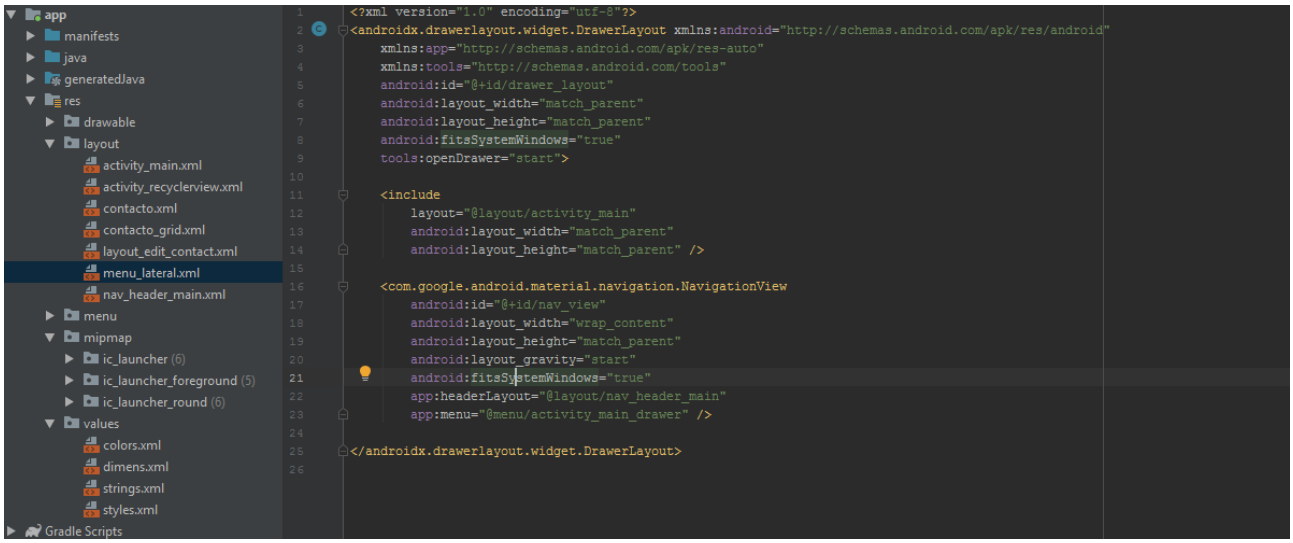

3:17
AgendaSqlite
1
1
2
3
44


3:18
AgendaSqlite
1
2
3
44
4v
5
Hola

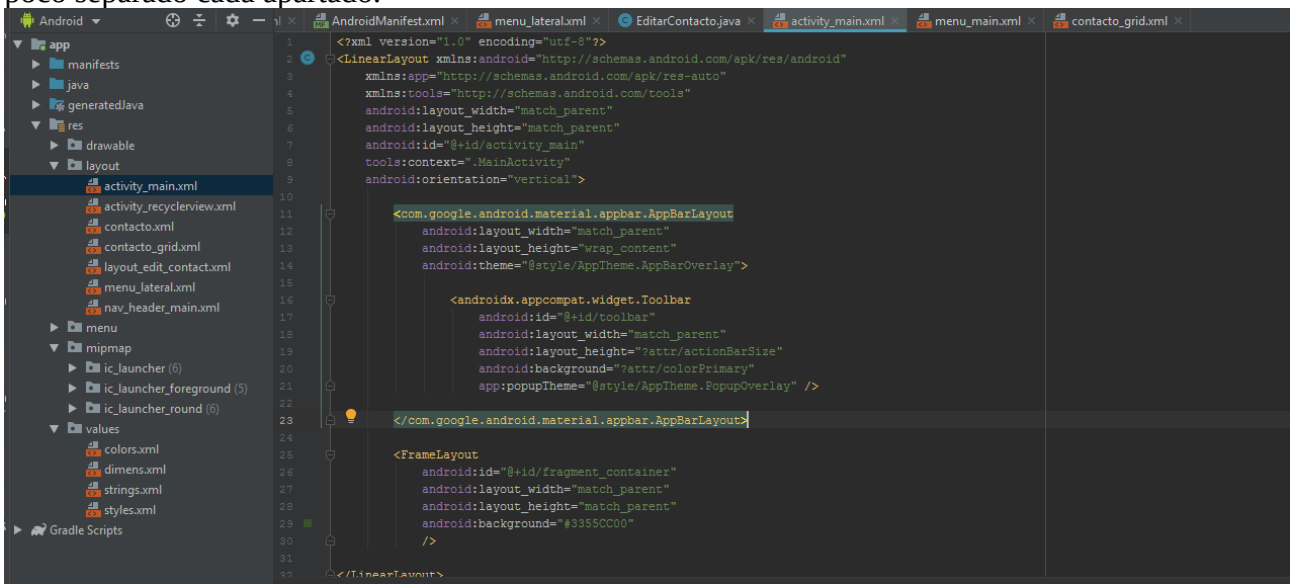

3:19
Agenda
Todos
Familias
Amigos
Trabajo
3
5

Vale, entonces para eso primero de todo tenemos que crear varios xml para que tenga un menu lateral.

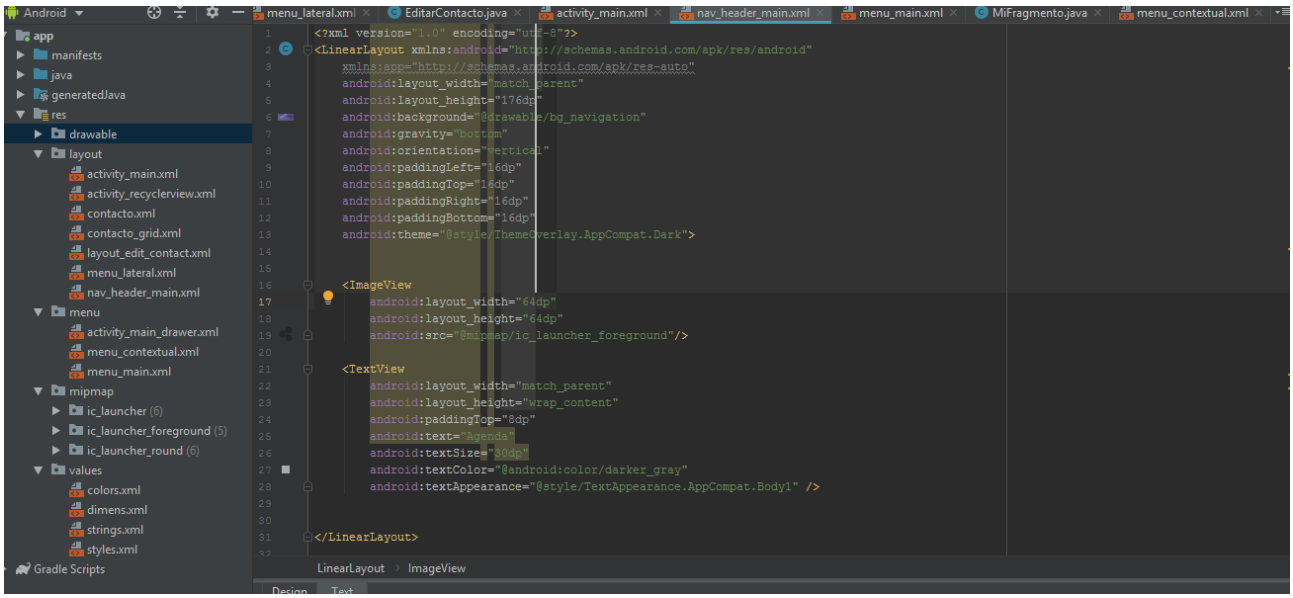
Empiezo con mi primer xml que es de DrawerLayout que tenga un un navigation view.



El mainActivity que he incluido es una actividad que va aparte para ver el aspecto más fácil. Un poco separado cada apartado.

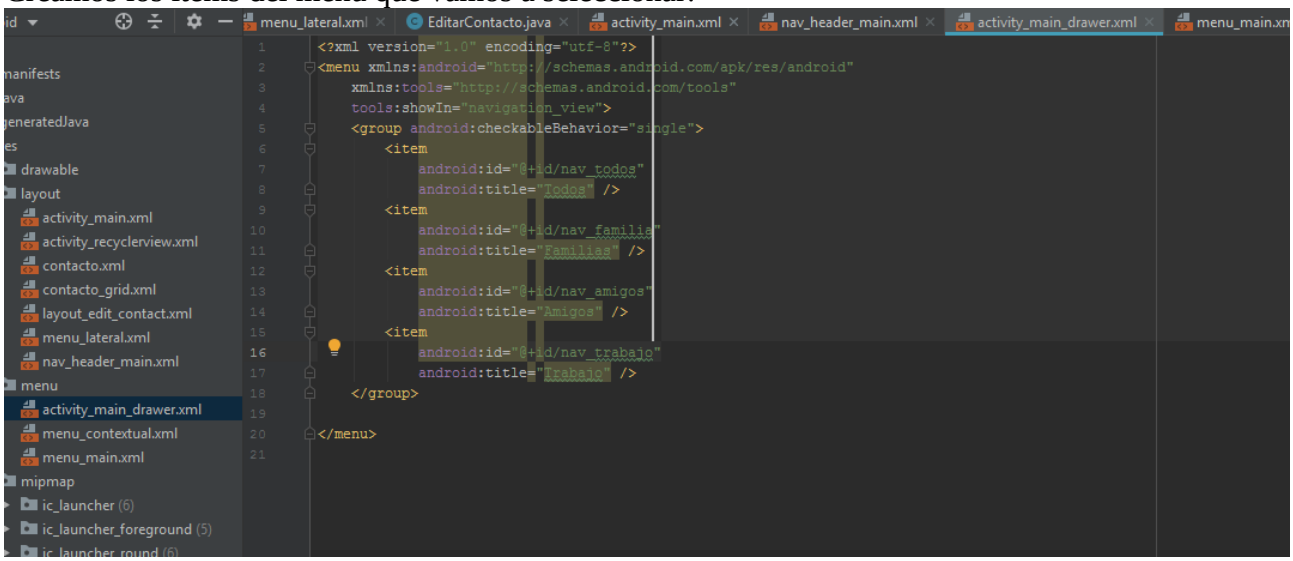


He cambiado a main activity añadiendo un Toolbar encima de el y lo he quitado el toolbar desde activity_recyclerview.xml.



Creamos un nav header que va a aparecer en nuestro panel del menú lateral. Creamos un textView y un image view y le ponemos un backgoud guapo.

Creamos los items del menu que vamos a seleccionar:



y para que aparezca un iconito al lado derecho de nuestra pantalla. Ponemos un menú demás para nuestros xml.

Y Ahora tendremos que programar para que nos aparezca el menú que hemos diseñado en xml. Y entonces vamos por la actividad principal y implementamos una interfaz ya hecha de navigation view para que podamos hacer que se conecte entre sí y conseguiremos que el menú funcioné como lo queramos.

```
ublic class MainActivity extends AppCompatActivity implements NavigationView.OnNavigationItemSelectedListener {
```

Y para que funcioné bien, hay que implementar los métodos sobreescritos.
El primer método que hacemos es el de navigation view. Porque la interfaz no obliga que implementamos el método. Este método le llega un id y es capaz de entender con que id queremos que se seleccione los datos.

```
@Override
public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
    // Handle navigation view item clicks here.
    int id = menuItem.getItemId();
    navigationItemSelectedId = id;

    Filtrar(id);

    DrawerLayout drawer = findViewById(R.id.drawer_layout);
    drawer.closeDrawer(GravityCompat.START);
    return true;
}

private void Filtrar(int id) {
    Cursor cur = null;
    switch (id){
        case R.id.nav_todos:
            break;
        case R.id.nav_amigos:
            cur = dbPersonas.rawQuery("SELECT * FROM personas where etiqueta LIKE '%"+"Amigos"+"%'",null);
            break;
        case R.id.nav_familia:
            cur = dbPersonas.rawQuery("SELECT * FROM personas where etiqueta LIKE '%"+"Familia"+"%'",null);
            break;
        case R.id.nav_trabajo:
            cur = dbPersonas.rawQuery("SELECT * FROM personas where etiqueta LIKE '%"+"Trabajo"+"%'",null);
            break;
    }
    CreaFragmento(cur);
```

Invalid VCS root mapp
The directory C:\Users

Así con la id me diferencio con las opciones. Y el filtro que tengo luego lo esplicaré con su tema de sqlite.
El filtro no funciona bien: lo dejo para final.

```java
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();


    //noinspection SimplifiableIfStatement
    if (id == R.id.Grid) {
        if(spanColumnas == 3) {
            spanColumnas = 1;
            ContactoLayout = R.layout.contacto;
        }
        else {
            spanColumnas = 3;
            ContactoLayout = R.layout.contacto_grid;
        }
        Filtrar(navigationItemSelectedId);
        return true;
    }

    return super.onOptionsItemSelected(item);
```

Y implemento dos métodos demás el primero que crea el grid del lado derecho de la pantalla del movil.

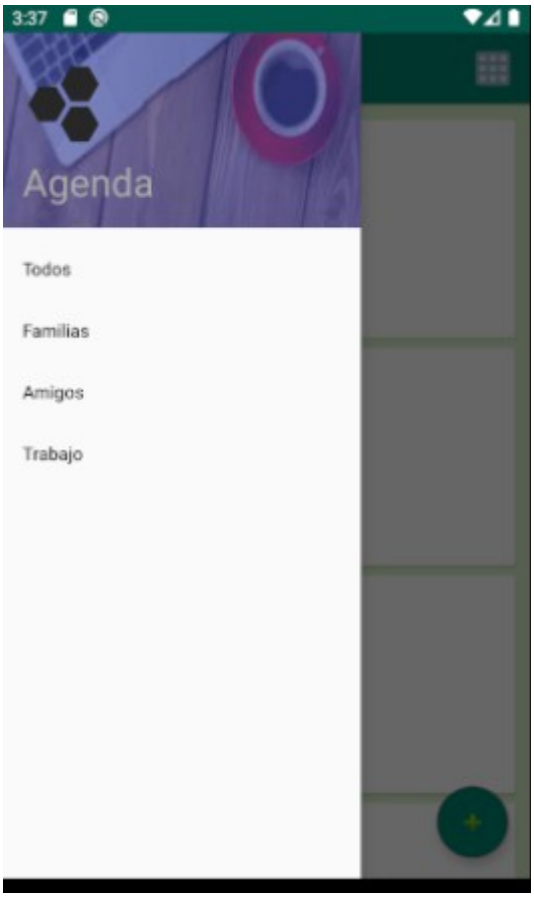
Y el segundo método es para seleccionar los filtros y le pongo un algorithmo que si el id es igual que el del grid del menú . Pues le cambio las columnas a 3. de aquí lo tengo cambio a 1 y el otro a 3 para mostrar un aspecto diferente.

Y por último Sobreescribimos el método backpressed().

```java
@Override
public void onBackPressed() {
    DrawerLayout drawer = findViewById(R.id.drawer_layout);
    if (drawer.isDrawerOpen(GravityCompat.START)) {
        drawer.closeDrawer(GravityCompat.START);
    } else {
        super.onBackPressed();
    }
}
```

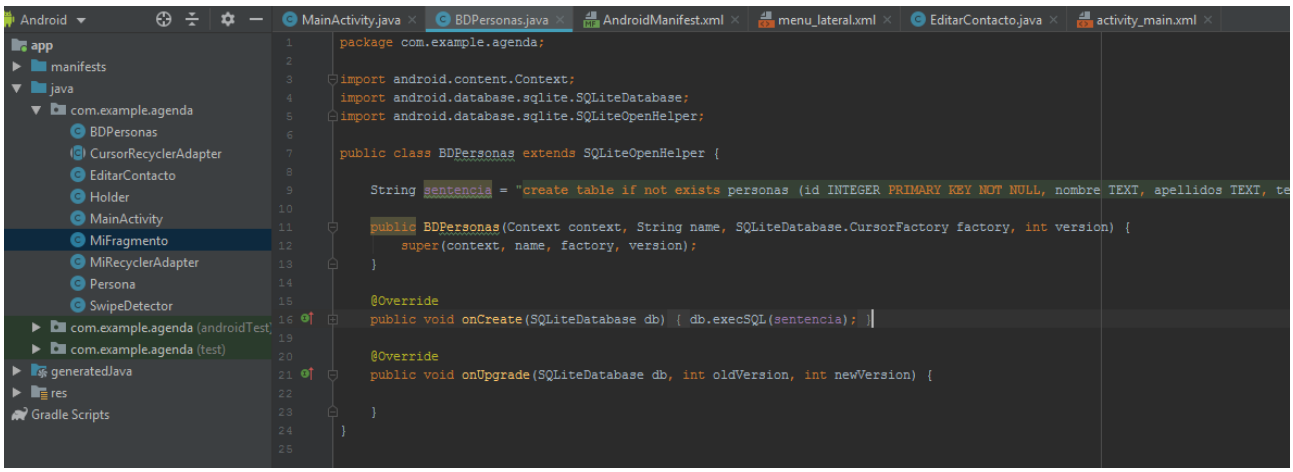
Y cambiamos el onCreate de MainActivity.

Y se quedará el aspecto así:



Y he añadidos varios variable y he cambiado el constructo de fragmento que le paso varias opciones.

Ya empezamos con el sqlite y primero de todos, creamos una clase que manejará de crear una base de datos y se crea una tabla.



Creamos la base de datos, ponemos el siguiente código.

```
BDPersonas personas;
SQLiteDatabase dbPersonas;
int ContactoLayout;

String imagenPorDefecto;
MiFragmento fragmentLista;
public RecyclerView recyclerView;
private int spanColumnas;
private int navigationItemSelectedId;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.menu_lateral);
    spanColumnas = 1;
    ContactoLayout = R.layout.contacto;

    personas = new BDPersonas(getBaseContext(), name: "BDPERSONAS", factory: null, version: 1);
    dbPersonas = personas.getWritableDatabase();
```

Para eso hay que utilizar el Adaptador creamos un adaptador que se extiende CursorRecyclerAdapter que a su vez que extiende de un Adapter que extiende de RecyclerView Adapter.

Creamos nuestro primer Adaptador en el que le pasamos todas las funciones que impleamentamos en Adaptador de recycler anteriormente.

```
import androidx.recyclerview.widget.RecyclerView;

public class MiRecyclerAdapter extends CursorRecyclerAdapter implements View.OnClickListener, View.OnTouchListener, View.OnLongClickListener  {

    private SQLiteDatabase sqLiteDatabase;
    private int mLayout;
    private Context c;
    View.OnClickListener listener;
    View.OnLongClickListener longClickListener;
    View.OnTouchListener onTouchListener;

    public MiRecyclerAdapter(int layout, Context c, Cursor cursor, SQLiteDatabase sqLiteDatabase) {
        super(c, cursor);
        this.sqLiteDatabase = sqLiteDatabase;
        mLayout = layout;
        this.c = c;
    }

    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext())
                .inflate(mLayout, parent, attachToRoot: false);
        view.setOnTouchListener(this);
        view.setOnClickListener(this);
        view.setOnLongClickListener(this);
        return new Holder(view,c);
    }
```

```java
public  void setClickOnView(View.OnClickListener listener) {
    if(listener!=null) this.listener= listener;
}
@Override
public void onClick(View view) { if(listener!=null) listener.onClick(view); }
public void SetOnTouchListener(View.OnTouchListener touchListener) {
    if(touchListener!=null) onTouchListener = touchListener;
}
@Override
public boolean onTouch(View view, MotionEvent motionEvent) {
    if(onTouchListener != null) {
        onTouchListener.onTouch(view, motionEvent);
    }
    return false;
}

public void  SetOnLongClick(View.OnLongClickListener longClickListener) {
    if(longClickListener != null) {
        this.longClickListener = longClickListener;
    }
}
@Override
public boolean onLongClick(View view) {
    if(longClickListener!=null) longClickListener.onLongClick(view);
    return false;
}
```

```java
@Override
public boolean onLongClick(View view) {
    if(longClickListener!=null) longClickListener.onLongClick(view);
    return false;
}

@Override
public void onBindViewHolder(RecyclerView.ViewHolder holder, Cursor cursor, int position) {

    int pos = position;

    cursor = sqLiteDatabase.rawQuery( sql: "SELECT * FROM personas",  selectionArgs: null);
    Cursor cur = sqLiteDatabase.rawQuery( sql: "SELECT * FROM personas where id=" + position,  selectionArgs: null);

    cur.moveToFirst();
    if(position < cursor.getCount()) {
        String nombre = cur.getString(cur.getColumnIndex( columnName: "nombre"));
        String apellidos = cur.getString(cur.getColumnIndex( columnName: "apellidos"));
        String telefono = cur.getString(cur.getColumnIndex( columnName: "telefono"));
        String correo = cur.getString(cur.getColumnIndex( columnName: "correo"));
        String imagenString = cur.getString(cur.getColumnIndex( columnName: "imagenString"));
        String etiquetaDesdeCampo = cur.getString(cur.getColumnIndex( columnName: "etiqueta"));

        String[] etiqueta = etiquetaDesdeCampo.split( regex: ",");

        Persona p = new Persona(nombre, apellidos, telefono, correo, imagenString, etiqueta);
        ((Holder)holder).bind(p);
    }
```

Igual que hicimos anteriormente ahora en este adapter, y le paso la layout, cursor , y el base de datos. Y Click listeners igual que antes.
Y en onBindviewholder(...) ejecuto las consultas ara sacar los datos.
Y con el cursor saco la persona que quiero mostrar en un cardview de mi recyclerView. Y le paso en un holder que tengo un método bind que recibe persona y lo coloca en sus lugares donde debe de colocar.

```java
@Override
public void onBindViewHolder(RecyclerView.ViewHolder holder, int position) {
    if(mCursor==null) {
        throw new IllegalStateException("Error, Cursor Vacío");
    }
    if(!mCursor.moveToPosition(position)) {
        throw new IllegalStateException("Error, no se puede encontrar la posicion: " + position);
    }
    onBindViewHolder(holder, mCursor, position);
}

public abstract void onBindViewHolder(RecyclerView.ViewHolder holder, Cursor cursor, int position);

@Override
public int getItemCount() {
    if(mCursor != null) {
        return  mCursor.getCount();
    }
    else
        return 0;
}
```

El el cursor recycler adapter recibe el contxeto y el cursor. Y en onCreateViewholder() le pasamos

```java
import androidx.recyclerview.widget.RecyclerView;


public abstract class CursorRecyclerAdapter extends RecyclerView.Adapter {

    Cursor mCursor;
    Context c;
    Holder h;

    public CursorRecyclerAdapter(Context c, Cursor cursor){
        this.c = c;
        mCursor = cursor;
    }

    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.contacto, parent,  attachToRoot: false);
        h = new Holder(view,c);
        return h;
    }

    @Override
    public void onBindViewHolder(RecyclerView.ViewHolder holder, int position) {
        if(mCursor==null) {
            throw new IllegalStateException("Error, Cursor Vacío");
        }
        if(!mCursor.moveToPosition(position)) {
            throw new IllegalStateException("Error, no se puede encontrar la posicion: " + position);
        }
        onBindViewHolder(holder, mCursor, position);
    }
```

el layout contacto y llamamos al holder.

```java
@Override
public long getItemId(int position) {
    if(hasStableIds() && mCursor != null) {
        if(mCursor.moveToPosition(position)) {
            mCursor.getLong(mCursor.getColumnIndexOrThrow( columnName: "id"));
        }
    }
    return RecyclerView.NO_ID;
}
```

Y creamos un método abstracto al que le llamamos desde MiRecyclerAdapter.

Y el método get item solamente devuelve la cantidad de los datos que tenga en base de datos.

Y luego creamos el holder de lo que estabamos hablando.
Y el holder es igual que antes.
Y su funcionamiento es igual que antes.

```
public class Holder extends RecyclerView.ViewHolder {

    Context context;
    View itemView;

    TextView txtNombre, txtApellido, txtTelefono, txtCorreo;
    ImageView imagen;
    View.OnClickListener listener;

    public Holder(View itemView, Context context) {
        super(itemView);
        this.context = context;
        this.itemView = itemView;
        txtNombre =  itemView.findViewById(R.id.Nombre);
        txtApellido= itemView.findViewById(R.id.Apellidos);
        txtCorreo = itemView.findViewById(R.id.Correo);
        txtTelefono = itemView.findViewById(R.id.Telefono);
        imagen = itemView.findViewById(R.id.imageView);

    }
    public  void bind(Persona persona){

        txtNombre.setText(persona.getNombre());
        txtApellido.setText(persona.getApellidos());
        txtCorreo.setText(persona.getCorreo());
        txtTelefono.setText(persona.getTelefono());
        imagen.setImageBitmap(MainActivity.StringABitmap(persona.getImagenString()));
    }

}
```

Tengo un método creaFragment que me sirve para actualizar la lista. Le paso el cursor y con la consuta que le digo. Y creamos fragments.

```
public void CreaFragmento(Cursor cursor) {
    BooraFragmento();
    fragmentLista = new MiFragmento( c this,cursor, personas,dbPersonas, imagenPorDefecto, recyclerView, spanColumnas);
    FragmentManager fragmentManager = getSupportFragmentManager();
    FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
    fragmentTransaction.add(R.id.fragment_container, fragmentLista);
    fragmentTransaction.commit();
}

private void BooraFragmento() {
    if(fragmentLista != null) {
        getSupportFragmentManager().beginTransaction().
                remove(getSupportFragmentManager().findFragmentById(R.id.fragment_container)).commit();
    }
}
```

```java
public class MiFragmento extends Fragment implements View.OnClickListener,  View.OnLongClickListener {

    private static final int CODIGO_EDITAR = 2;
    private static final int CODIGO_AÑADIR = 3;
    public RecyclerView recyclerView;
    MiRecyclerAdapter mAdapter;
    private SwipeDetector swipeDetector;
    SQLiteDatabase sqLiteDatabase;
    BDPersonas dbPersonas;
    Cursor miCursor;
    ImageView imagen;
    String imagenPorDefecto;
    private View rootView;
    int pos;
    Persona p;
    Context context;
    public int spanCount;

    public MiFragmento(Context c, Cursor cursor,BDPersonas personas,SQLiteDatabase datos, String imagenPorDefecto, RecyclerView recyclerView, int colu
        this.sqLiteDatabase = datos;
        this.imagenPorDefecto = imagenPorDefecto;
        this.recyclerView = recyclerView;
        this.dbPersonas = personas;
        this.spanCount = columnas;
        if(cursor != null) {
            this.miCursor = cursor;
        } else {
            miCursor = sqLiteDatabase.rawQuery( sql: "select * from personas",  selectionArgs: null);
        }
        context = c;
```

Ya mi fragmento tiene cambiado el codigo. Vamos a hacer pequeños cambios para que tenga un sentido de lo que estamos haciendo.

```java
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    super.onCreateView(inflater, container, savedInstanceState);
    rootView = inflater.inflate(R.layout.activity_recyclerview, container,  attachToRoot: false);
    imagen = rootView.findViewById(R.id.imageView);

    FloatingActionButton fab = rootView.findViewById(R.id.fab);
    fab.setOnClickListener((view) → { AñadirDatos(); });

    sqLiteDatabase = dbPersonas.getReadableDatabase();
    if(sqLiteDatabase != null) {
        recyclerView = rootView.findViewById(R.id.recycler);
        CargarRecyclerView();
    }
    return rootView;
}

private void CargarRecyclerView() {
    recyclerView.setHasFixedSize(true);
    mAdapter = new MiRecyclerAdapter(((MainActivity)context).ContactoLayout, getContext(), miCursor, sqLiteDatabase );
    recyclerView.setAdapter(mAdapter);
    recyclerView.setLayoutManager(new GridLayoutManager(getActivity(), spanCount));
    mAdapter.SetOnLongClick( longClickListener: this);
    swipeDetector = new SwipeDetector();
    mAdapter.setClickOnView(this);
    mAdapter.SetOnTouchListener(swipeDetector);
}
```

El código es igual, solamente vamos cambiando poco a poco. Cargando el recycler.

```
@Override
public boolean onLongClick(View v) {
    pos = recyclerView.getChildAdapterPosition(v);

    String nombre = SeleccionarCampoSelect( sentencia: "select * from personas where id = " + pos,  campo: "nombre");

    AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
    builder.setMessage("¿Seguro que quieres eliminar a " + nombre + "?");
    builder.setPositiveButton( text: "ELIMINAR", (dialog, which) → {

            String[] args = {Integer.toString(pos)};
            sqLiteDatabase.delete( table: "personas",  whereClause: "id=?", args);

            sqLiteDatabase.execSQL("UPDATE personas SET id=id-1 WHERE id > " + pos);

            ((MainActivity)context).CreaFragmento(sqLiteDatabase.rawQuery( sql: "select * from personas",  selectionArgs: null));
    });
    builder.setNegativeButton( text: "CANCELAR", (dialog, which) → {
            dialog.cancel();
    });
    builder.create().show();
    return true;
}
```

Y para sacar el nombre he creado un método que recibe la sentencia pero, en el segundo parámetro se recibe el “nombre” del campo para sacar el primer nombre que se encuentra.
Así se pueda sacar cualquier otro campo igualmente y ahorramos el código.

```
public String SeleccionarCampoSelect(String sentencia, String campo) {
    sqLiteDatabase = dbPersonas.getReadableDatabase();
    String campoString = "";
    if(sqLiteDatabase != null) {
        Cursor cursor = sqLiteDatabase.rawQuery(sentencia,  selectionArgs: null);
        if(cursor != null) {
            cursor.moveToFirst();
            campoString = cursor.getString(cursor.getColumnIndex(campo));
        }
    }
    return campoString;
}
```

Y ahora el metodo de on long click que teniamos en nuestro fragment sacamos el nombre desde el cursor y ponemos en el dialogo para mostrar lo que estamos borrando.

```
@Override
public boolean onLongClick(View v) {
    pos = recyclerView.getChildAdapterPosition(v);

    String nombre = SeleccionarCampoSelect( sentencia: "select * from personas where id = " + pos,  campo: "nombre");

    AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
    builder.setMessage("¿Seguro que quieres eliminar a " + nombre + "?");
    builder.setPositiveButton( text: "ELIMINAR", (dialog, which) → {

            String[] args = {Integer.toString(pos)};
            sqLiteDatabase.delete( table: "personas",  whereClause: "id=?", args);

            sqLiteDatabase.execSQL("UPDATE personas SET id=id-1 WHERE id > " + pos);

            ((MainActivity)context).CreaFragmento(sqLiteDatabase.rawQuery( sql: "select * from personas",  selectionArgs: null));
    });
    builder.setNegativeButton( text: "CANCELAR", (dialog, which) → {
            dialog.cancel();
    });
    builder.create().show();
    return true;
}
```

Igual sacamos los datos desde el método creado para sacar cadenas del campo de cada uno para mostrar al usuario en el diialogo.

```
public void onClick(View v) {
    pos = recyclerView.getChildAdapterPosition(v);

    final String telefono = SeleccionarCampoSelect( sentencia: "select * from personas where id = " + pos,  campo: "telefono");
    String nombre = SeleccionarCampoSelect( sentencia: "select * from personas where id = " + pos,  campo: "nombre");
    final String correo = SeleccionarCampoSelect( sentencia: "select * from personas where id = " + pos,  campo: "correo");

    if (swipeDetector.swipeDetected()) {
        AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
        switch (swipeDetector.getAction()) {
            case LR:
                builder.setMessage("¿Estas seguro que quieres llamar a " + nombre + "?");
                builder.setPositiveButton( text: "LLAMAR", (dialog, which) -> {
                        Intent intent = new Intent(Intent.ACTION_DIAL);
                        intent.setData(Uri.parse("telefono: " + telefono));
                        if (intent.resolveActivity(getContext().getPackageManager()) != null) {
                            startActivity(intent);
                        }
                });
                builder.setNegativeButton( text: "Cancelar", (dialog, which) -> {
                        dialog.cancel();
                });
                builder.create().show();
                break;
            case RL:
                builder = new AlertDialog.Builder(getContext());
                builder.setMessage("¿Estas seguro que quieres enviar mensaje a " + nombre + "?");
                builder.setPositiveButton( text: "enviar", (dialog, which) -> {
                        Intent intent = new Intent(Intent.ACTION_SENDTO);
                        intent.setData(Uri.fromParts( scheme: "mailto", correo,  fragment: null));
                        Intent chooser = Intent.createChooser(intent,  title: "Enviar mensaje...");
```

Claro estamos con la base de datos interna del movil SQLite, pues que tenemos que guardar los datos para mostrar los. Entonces:
Guardamos los datos cuando recibamos desde la activity de Editar contacto, lo que ya esta explicado anteriormente. Pero he hecho algún cambio y luego lo muestro para que se quede todo cuadrado.
Para insertar y para añadir utilizamos así:
Este es el método onActivityResul(): ya que tengo explicado anteriormente. Por eso solo se hace un poquito de modificaciones y ya.

```
    if (resultCode == RESULT_OK) {
        p = data.getParcelableExtra( name: "PersonaEditado");
        if(p != null){

            ContentValues valores = new ContentValues();
            valores.put("nombre", p.getNombre());
            valores.put("apellidos", p.getApellidos());
            valores.put("telefono", p.getTelefono());
            valores.put("correo", p.getCorreo());
            valores.put("imagenString", p.getImagenString());
            valores.put("etiqueta", p.getEtiqueta()[0] +", "+p.getEtiqueta()[1]+", "+p.getEtiqueta()[2]);
            String[] args = {Integer.toString(pos)};
            sqLiteDatabase.update( table: "personas",valores,  whereClause: "id=?", args);
        }
    }
    break;
case CODIGO_AÑADIR:
    if (resultCode == RESULT_OK) {
        p = data.getParcelableExtra( name: "PersonaEditado");
        if(p != null){
            miCursor = sqLiteDatabase.rawQuery( sql: "select * from personas",  selectionArgs: null);
            ContentValues valores = new ContentValues();
            valores.put("id", miCursor.getCount());
            valores.put("nombre", p.getNombre());
            valores.put("apellidos", p.getApellidos());
            valores.put("telefono", p.getTelefono());
            valores.put("correo", p.getCorreo());
            valores.put("imagenString", p.getImagenString());
            valores.put("etiqueta", p.getEtiqueta()[0] +", "+p.getEtiqueta()[1]+", "+p.getEtiqueta()[2]);
            String[] args = {Integer.toString(pos)};
            sqLiteDatabase.insert( table: "personas",  nullColumnHack: null, valores);
```

```
private void EditarDatos() {
    Intent intent = new Intent(getContext(), EditarContacto.class);
    String sentencia = "SELECT * FROM personas WHERE id=" + pos;
    String nombre = SeleccionarCampoSelect(sentencia, campo: "nombre");
    String apellidos = SeleccionarCampoSelect(sentencia, campo: "apellidos");
    String telefono = SeleccionarCampoSelect(sentencia, campo: "telefono");
    String correo = SeleccionarCampoSelect(sentencia, campo: "correo");
    String imagenString = SeleccionarCampoSelect(sentencia, campo: "imagenString");
    String etiquetaDesdeCampo = SeleccionarCampoSelect(sentencia, campo: "etiqueta");

    String[] etiqueta = etiquetaDesdeCampo.split( regex: ",");

    Persona p = new Persona(nombre, apellidos, telefono, correo, imagenString, etiqueta);

    intent.putExtra( name: "DatoPersona", p);
    startActivityForResult(intent,CODIGO_EDITAR);
}
private void AñadirDatos() {
    Intent intent = new Intent(getContext(), EditarContacto.class);
    Persona p = null;
    intent.putExtra( name: "DatoPersona", p);
    startActivityForResult(intent,CODIGO_AÑADIR);
}
```

Para editar datos primero me lo recupero los datos de la base de datos y lo paso a la actividad para cammbiarlo y con el método anterior de OnActivityResult() lo recuperamos y cambiamos los datos que queramos cambiar. O igual para añadir datos mandamos a la actividad editar contacto que se crea un nuevo usuario y pasa los datos de la misma y inserta datos Con OnActivityOnResult() y guarda la base de datos.

En la actividad Editar contacto he puesto un menú contextual a la imagen que era pordefecto. Ya podemos elegir para tomar, guardar foto, dejar por defecto o cancelar la operación.
Esta clase es igual que la clase que hisimos nosotros pero resulta que cambio aquí algunas cosillas. Voy diciendo poco a poco.
La clase persona añadimos un campo nuevo porque necesitamos guardar las etiquetas de cada usuario. Ya que he dicho que no funcionaba bien las etiquetas. Lo dejo por ahora pero lo tendremos en cuenta para luego.

```
    private static final int CODIGO_GALERIA = 1;
    private static final int REQUEST_IMAGE_CAPTURE = 2;

    private Persona p = null;
    private ImageView imagen;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.layout_edit_contact);

        final EditText editTextNombre = findViewById(R.id.editText);
        final EditText editTextApellidos = findViewById(R.id.editText2);
        final EditText editTextTelefono = findViewById(R.id.editText3);
        final EditText editTextCorreo = findViewById(R.id.editText4);
        final CheckBox checkBoxAmigos = findViewById(R.id.checkBoxAmigos);
        final CheckBox checkBoxFamilia = findViewById(R.id.checkBoxFamilia);
        final CheckBox checkBoxTrabajo = findViewById(R.id.checkBoxTrabajo);



        imagen = findViewById(R.id.imagen);
        imagen.setOnClickListener(this);
        FloatingActionButton fab = findViewById(R.id.floatingActionButton);

        Intent intentR = getIntent();
        p = intentR.getParcelableExtra( name: "DatoPersona");
```

Colocamos cada objeto a su lugar.

```
if(p!= null){

    editTextNombre.setText(p.getNombre());

    editTextApellidos.setText(p.getApellidos());

    editTextTelefono.setText(p.getTelefono());

    editTextCorreo.setText(p.getCorreo());

    String ArrayImagen = p.getImagenString();

    String[] etiquetas = p.getEtiqueta();

    if(etiquetas[0] == "Familia")
        checkBoxFamilia.setChecked(true);
    if(etiquetas[1] == "Amigos")
        checkBoxAmigos.setChecked(true);
    if(etiquetas[2] == "Trabajo")
        checkBoxTrabajo.setChecked(true);

    imagen.setImageBitmap(MainActivity.StringABitmap(ArrayImagen));
    registerForContextMenu(imagen);

}
else {
    p = new Persona();
    imagen.setImageBitmap(Bitmap.createScaledBitmap(BitmapFactory.de
}
```

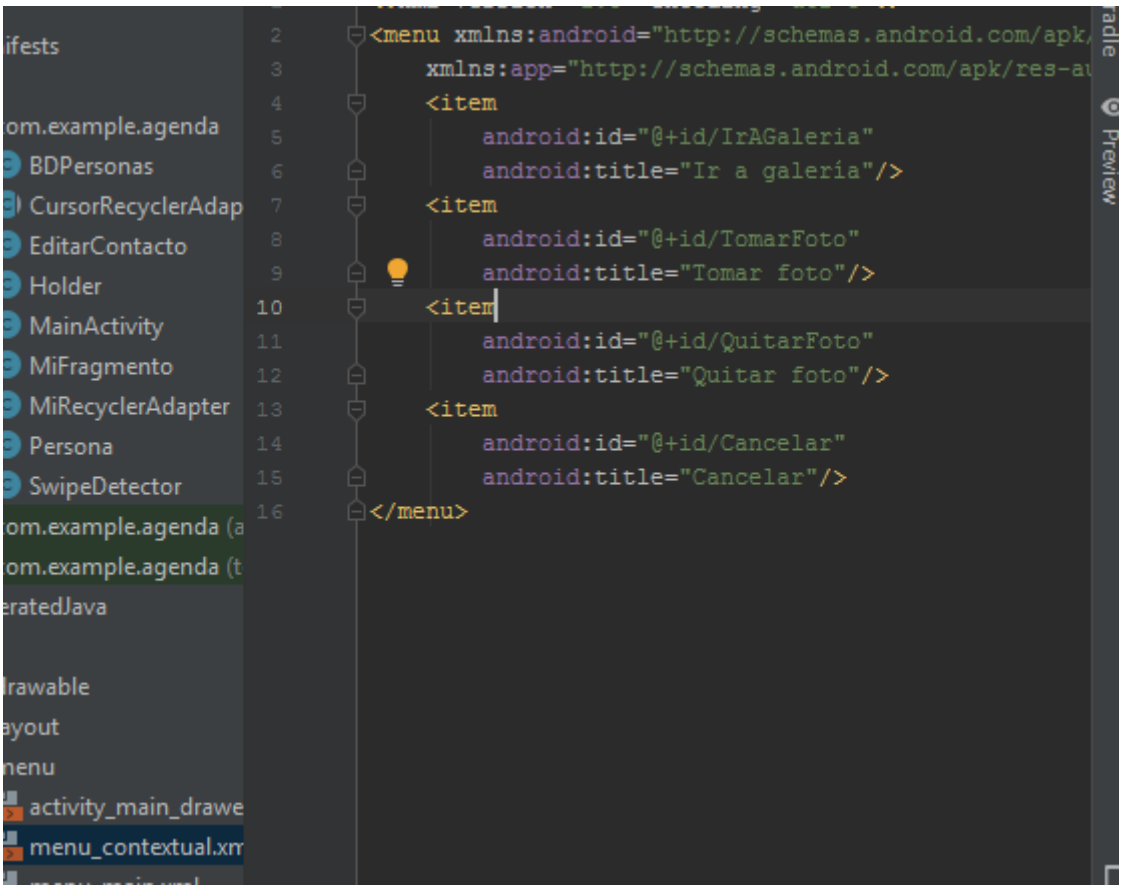
Intento que se quede checkeado cuando se encuentra la etiqueta pero no se no me funciona.

```
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        Intent intent = new Intent(getApplicationContext(), MainActivit
        p.setNombre(editTextNombre.getText().toString());
        p.setApellidos(editTextApellidos.getText().toString());
        p.setCorreo(editTextCorreo.getText().toString());
        p.setTelefono(editTextTelefono.getText().toString());
        String[] etiquetas = new String[3];
        if(checkBoxFamilia.isChecked())
            etiquetas[0] = "Familia";
        if(checkBoxAmigos.isChecked())
            etiquetas[1] = "Amigos";
        if(checkBoxTrabajo.isChecked())
            etiquetas[2] = "Trabajo";
        p.setEtiqueta(etiquetas);

        BitmapDrawable drawable = (BitmapDrawable) imagen.getDrawable()
        if(drawable != null) {
            Bitmap bitmap = drawable.getBitmap();
            p.setImagenString(MainActivity.BitmapAString(bitmap));
        }
        else
            p.setImagenString(MainActivity.BitmapAString(BitmapFactory.
```

El fab es igual solamente que le he cambiado por las etiquetas. Y al final igual que antes.

```
if(!p.getNombre().equals("")){
    intent.putExtra( name: "PersonaEditado", p);
    setResult(RESULT_OK, intent);
    finish();
}
else
    Toast.makeText(getApplicationContext(), text: "El nombre es
```

El menú contextual que le he puesto a la imagen aquí tengo su xml:



para que funcione el menú contextual tendremos que poner el siguiente código en onCreate de EditarContacto:

```
registerForContextMenu(imagen);
```

Código para ir a la galería:

```
}
public void IrAGaleria() {
    Intent intent = new Intent();
    intent.setType("image/*");
    intent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(
            Intent.createChooser(intent, title: "Seleccione una imagen"),
            CODIGO_GALERIA);
}
```

Código para tomar la foto:

```
private void TomarFoto() {
    Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);
    }
}
```

Y después de tomar o traer foto desde la galería tenemos que leer en nuestra actividad:

```
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    Uri selectedImage;

    switch (requestCode) {
        case CODIGO_GALERIA:
            if (resultCode == RESULT_OK) {
                selectedImage = data.getData();
                String selectedPath = selectedImage.getPath();
                if (requestCode == 1) {
                    if (selectedPath != null) {
                        InputStream imageStream = null;
                        try {
                            imageStream = getContentResolver().openInputStream(
                                    selectedImage);
                        } catch (FileNotFoundException e) {
                            e.printStackTrace();
                        }
                        Bitmap bmp = BitmapFactory.decodeStream(imageStream);
                        imagen.setImageBitmap(Bitmap.createScaledBitmap(bmp,  dstWidth: 100,  dstHeight: 100,  filter: true));
                    }
                }
            }
            break;
            case REQUEST_IMAGE_CAPTURE:
                if (resultCode == RESULT_OK) {
                    Bundle extras = data.getExtras();
                    Bitmap imageBitmap = (Bitmap) extras.get("data"); //"destinoFoto" es el imageView seleccionad en la APP para mostrar la fotografía tomada.
                    imagen.setImageBitmap(Bitmap.createScaledBitmap(imageBitmap,  dstWidth: 100,  dstHeight: 100,  filter: true));
                }
                break;
    }
```

y para que cada campo del menú funcione separado y correctamente, tendremos que implementar un método Sobrescrito. Que se diferencia entre los campos del menú.

```
@Override
public boolean onContextItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.IrAGaleria:
            IrAGaleria();
            break;
        case R.id.TomarFoto:
            TomarFoto();
            break;
        case R.id.QuitarFoto:
            imagen.setImageBitmap(Bitmap.createScaledBitmap(BitmapFactory.decodeResource(this.getResources(), R.drawable.pordefecto_imagen),  dstWidth: 100,  dstHeight: 100,  filter: t
            BitmapDrawable drawable = (BitmapDrawable) imagen.getDrawable();
            if(drawable != null) {
                Bitmap bitmap = drawable.getBitmap();
                p.setImagenString(MainActivity.BitmapAString(bitmap));
            }
            break;
        case R.id.Cancelar:
            break;
    }
```

Y todo los de más va igual que antes.

Un Saludo,
Mahroz Jawad.
Gracia!